Sveučilište u Zagrebu

Fakultet strojarstva i brodogradnje

ZAVRŠNI RAD

Ivan Filipović

Zagreb, 2016

Sveučilište u Zagrebu

Fakultet strojarstva i brodogradnje

ZAVRŠNI RAD

STUDENT:
Ivan Filipović

MENTOR:
Prof. dr. sc. Mladen Crneković

Zagreb, 2016

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE
Središnje povjerenstvo za završne i diplomske ispite
Povjerenstvo za završne ispite studija strojarstva za smjerove:
proizvodno inženjerstvo, računalno inženjerstvo, industrijsko inženjerstvo i menadžment, inženjerstvo materijala i mehatronika i robotika

Sveučilište u Zagrebu Fakultet strojarstva i brodogradnje	
Datum	Prilog
Klasa:	
Ur.broj:	

ZAVRŠNI ZADATAK

Student: IVAN FILIPOVIĆ Mat. br.: 0035188244

Naslov rada na hrvatskom jeziku: **DETEKTOR PREPREKA MOBILNOG ROBOTA**

Naslov rada na engleskom jeziku: **OBSTACLE DETECTOR FOR MOBILE ROBOT**

Opis zadatka:

Gibanje mobilnog robota ovisno je o njegovoj percepciji okoline u kojoj se kreće. Percepcija okoline robota ostvaruje se odgovarajućim senzorima i obradom signala prema očekivanom modelu okoline. Ako je upravljački sustav manje procesorske moći (npr. 8 bitni kontroler) obrada velike količine podataka (npr. signal s kamere) nije moguća. Zato se koriste jednostavniji senzori kao što su laserski, ultrazvučni i infracrveni.

Od infracrvenog daljinomjera potrebno je konstruirati samostalni snimač okoline (engl. scaner) koji bi obuhvaćao kut od 180°, a podatke prenosio nadređenom računalu. Konstrukcija snimača treba biti takva da se može montirati na mobilnog robota.

Tražena rješenja:

- odabir i obrazloženje komponenata snimača,
- program upravljanja,
- izrada i testiranje kompaktnog modela s prikazom rezultata na osobnom računalu.

Zadatak zadan:

30. studenog 2016.

Zadatak zadao:

Prof.dr.sc. Mladen Crneković

Rok predaje rada:
1. rok: 24. veljače 2017.
2. rok (izvanredni): 28. lipnja 2017.
3. rok: 22. rujna 2017.

Predviđeni datumi obrane:
1. rok: 27.2. - 03.03. 2017.
2. rok (izvanredni): 30. 06. 2017.
3. rok: 25.9. - 29. 09. 2017.

v.d. predsjednik Povjerenstva:

Izv. prof. dr. sc. Branko Bauer

Izjavljujem da sam ovaj rad izradio samostalno koristeći stečena znanja tijekom studija i navedenu literaturu.

Zahvaljujem se svom mentoru Prof. dr. sc. Mladenu Crnekoviću na pristupačnosti i strpljenju, savjetima i pomoći pri izradi ovoga završnog rada.

Ivan Filipović

Sadržaj:

Tablica slika:

Tablice:

STRANICA ZA ZADATAK

1 Uvod

Zadatak ovog rada je riješiti problem detektiranja prepreka kod mobilnih robota s upravljačkim sustavom malih procesorskih moći. Kao što je u samom zadatku spomenuto, obrada slike preko kamere nije moguća pa se problem svodi na riješenje pomoću infracrvenog daljinomjera. Ideja rada je napraviti detektor prepreka koji će obuhvaćati kut od 180° te podatke slati na računalo gdje će se isti prikazati na grafičkom sučelju prihvatljivo ljudskom oku.

Sama izvedba se sastoji od konstrukcije nosača na kojemu se nalaze dva infracrvena daljinomjera tvrtke SHARP modela 2Y0A21 koji se nalaze pod kutem 60° jedan od drugoga, električni servo motor modela TOWARDPRO MG996R sa mogućnošću zakreta od 120° te ARDUINO NANO mikrokontrolerom kao upravljačkom jedinicom.

Na tržištu se mogu naći i senzori udaljenosti bazirani na drugim tehnologijama, kao na primjer laserski daljinomjeri tvrtke PEPERL+FUCHS s mogućnošću raznih udaljenosti na kojima mogu detektirati predmete, model OMT100-R100-EP [4] s mjernim dometom od 40 do 100 mm ili VDM18-300/21/122/151 [5] s dometom od 80 do 300 mm. Isto tako postoje senzori bazirani na sonaru. Primjer takvoga daljinomjera je PARRALAX-ov Ultrasonic Distance Sensor #28015 [6] koji ima mjerni domet od 2 cm do 3m. Zbog cijene i pristupačnosti su za ovaj rad odabrani SHARP-ovi senzori iako bi po nekim drugim kriterijima malo skuplji senzori bili bolji (mjerna udaljenost, razlučivost). U izradi ovog zadatka odabrani senzori su dobro riješavali zadani problem.

Programiranje Arduino Nano upravljačke jedinice je izvedeno u programskom jeziku Arduino, a grafički prikaz u programu Processing. Dva spomenuta programa komuniciraju preko serijske veze.

Cilj rada je mjerene podatke sa senzora prebaciti na računalo i prikazati ih grafički te na taj način dobiti pogled na prostor i prepreke ispred senzora ili u slučaju ugradnje na mobilnog robota detektirati prepreke ispred samog robota.

2 Komponente

Uređaj se sastoji od sljedećih komponenti:

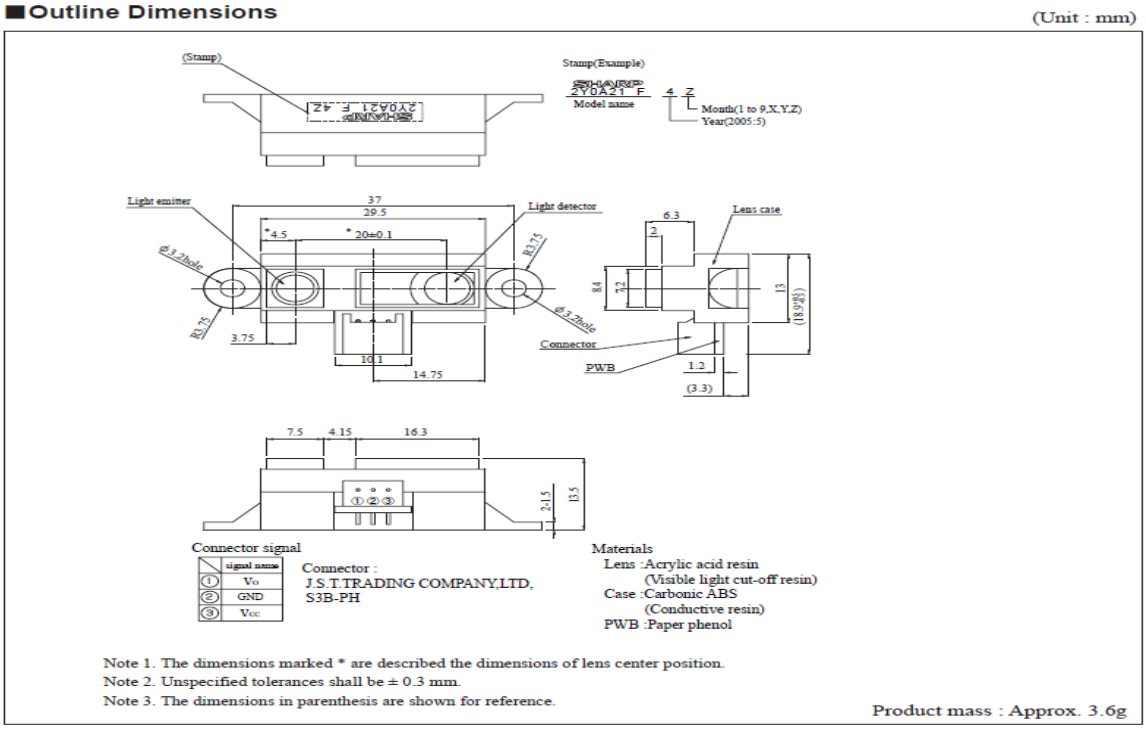
- Infracrveni daljinomjer SHARP 2Y0A21 x2
- Električni servo motor TOWARDPRO MG 996R
- ARDUINO NANO
- Konstrukcija nosača

2.1 Infracrveni daljinomjer SHARP 2Y0A21

2Y0A21 je daljinomjer sastavljen od integrirane kombinacije PSD-a (position sensitive detector/ osjetljivi detektor pozicije), IRED-a (infrared emitting diode/infracrvene emitirajuče diode) i sklopa za procesiranje signala. Reflektivnost objekta, temperatura okoline i trajanje operacije vrlo malo utječu na mjeru udaljenosti zbog adaptivne metode trianguliranja. Uređaj kao izlazni signal ima napon proporcionalnu udaljenosti koju mjeri te se koristi kao senzor daljine.

Značajke senzora

- Mjerni domet: 10 do 80 cm
- Analogni izlaz
- Dimenzije senzora: 29.5x13x13.5 mm
- Potrošnja struje : 30 mA
- Potreban napon : 4.5 do 5.5 V

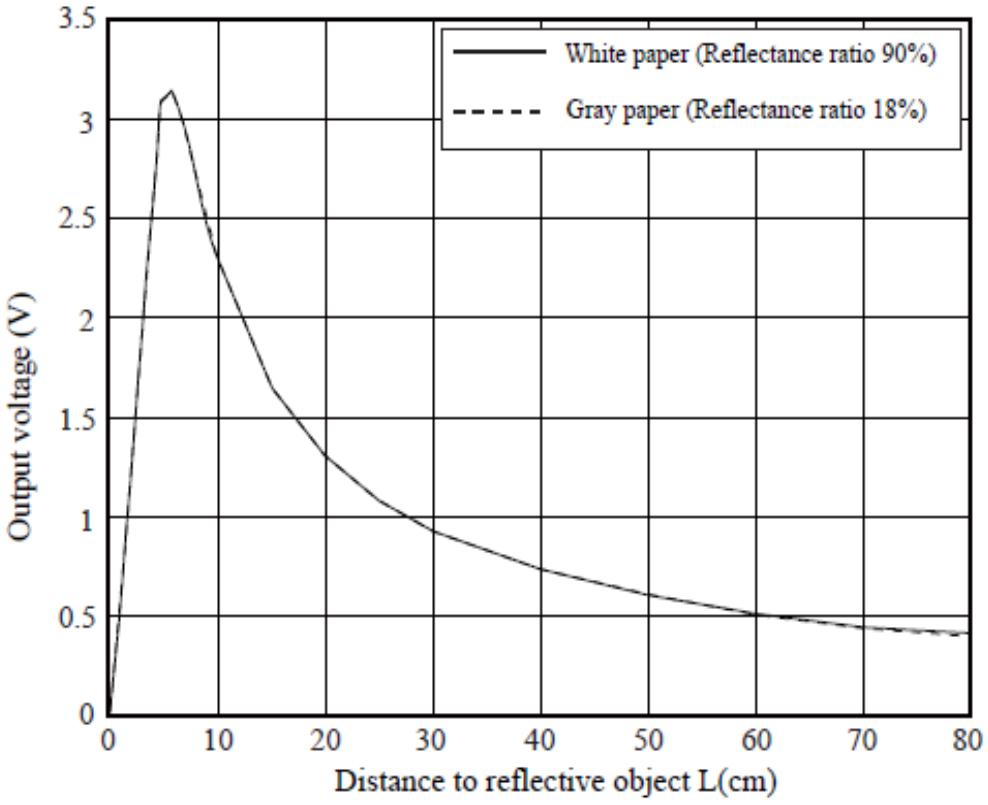


Slika 1. Dimenzije senzora SHARP 2Y0A21 [1]

Elektro optičke značajke

Tablica 1 (značajke senzora):

Parametar	Simbol	Uvijet	MIN.	TYP.	MAX.	Mjerna jedinica
Prosječna struja napajanja	I_{cc}	L=80cm	-	30	40	mA
Mjerena udaljenost	ΔL		10	-	80	cm
Izlazna voltaža	V_O	L=80cm	0.25	0.4	0.55	V
Razlika izlazne voltaže	ΔV_O	Razlika voltaže između L=10cm i L=80cm	1.65	1.9	2.15	V



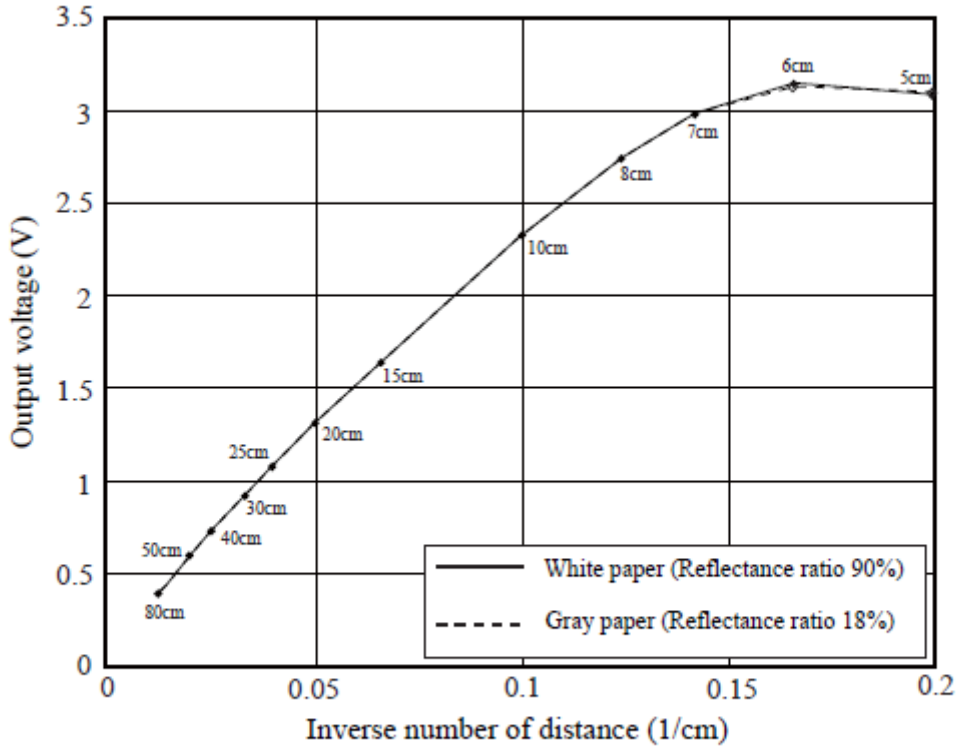
Slika 2. Odnos napona i udaljenosti (karakteristika senzora) [1]

Na slici 2 prikazan je odnos udaljenosti mjerenog predmeta i izlaznog napona senzora. Na karakteristici vidimo linearni uspon krivulje u vrijednostima od 0 do 7 cm. Taj dio karakteristike se odbacuje zbog male rezolucije udaljenosti u odnosu na izlazni napon te zbog preklapanja vrijednosti izlaznog napona sa udaljenosti, na primjer na izlaznom naponu 2 V se mogu dogoditi dvije vrijednosti udaljenosti, 3 i 12, stoga se prvi linearni dio karakteristike odbacuje i uzima se samo krivulja do 10 do 80 cm udaljenosti, što je vidljivo i iz tablice 1.

Inverz mjerene udaljenosti je relativno linearan u području mjernog intervala senzora. Zbog ove linearizacije možemo lako dobiti funkciju koja povezuje analogni izlazni napon i mjerenu udaljenost tako da konstantu 23.61 V*cm dijelimo s dobivenim izlaznim naponom. Na to se još može dodati i konstanta s kojom se uspravljaju greške u samom senzoru u svrhu dobivanja što točnijeg rezultata. U primjeru riješenom u ovom zadatku formula korištena za što bolju povezanost izlaznog napona senzora i mjerene udaljenosti glasi :

$$\frac{23.61}{volt1 - 0.1696} \cdot 1000$$

Gdje je *volt1* mjereni napon, 0.1696 konstanta greške a 23.61 konstanta senzora.



Slika 3. Odnos napona i udaljenosti (karakteristika senzora) [1]

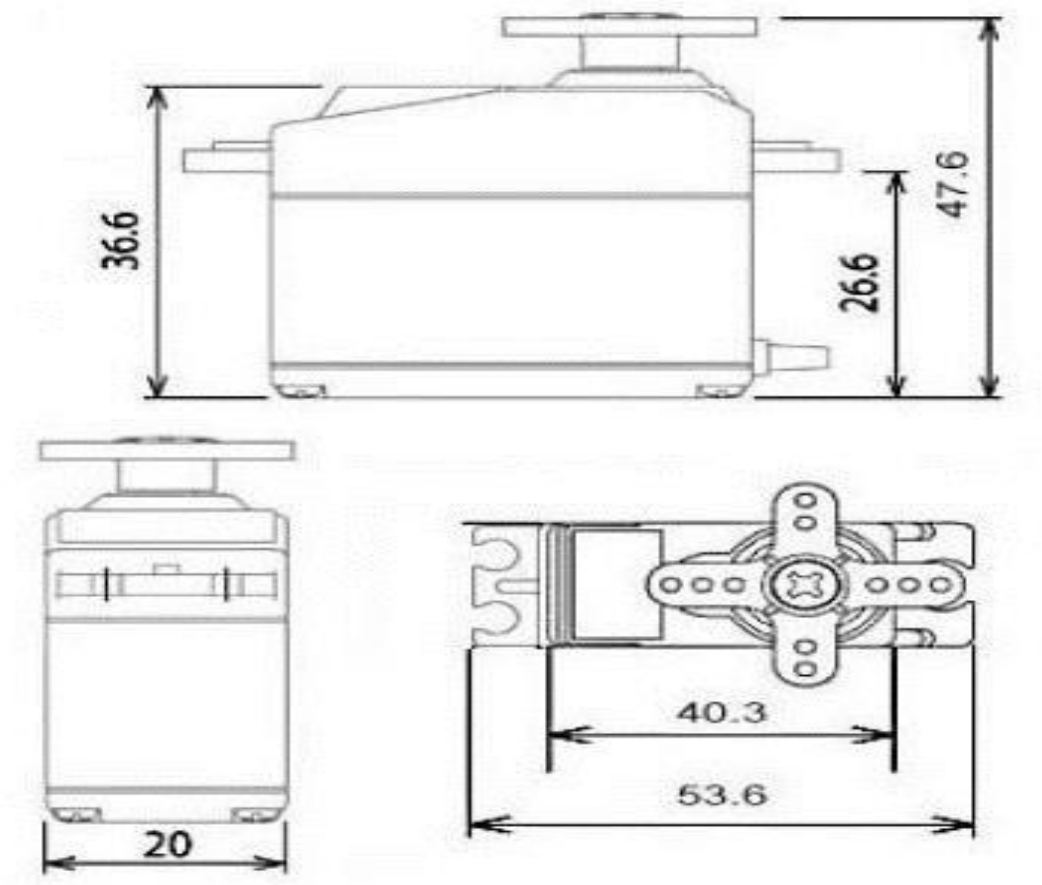
2.2 Električni servomotor TOWARDPRO MG 996R

MG 996R je digitalni servo elektromotor sa metalnim zupčanicima. Ima mogućnost zakretanja od 120° (60°u svakom smjeru). Kao ulazni signal koristi PWM(pulsno širinsku modulaciju) gdje jedan signal traje 20ms a napon potreban za to mu je od 4.8V do 7.2V. Mogućnost preciznog pozicioniranja i male dimenzije čini ga prihvatljivim za ovaj projekt.

Specifikacije motora

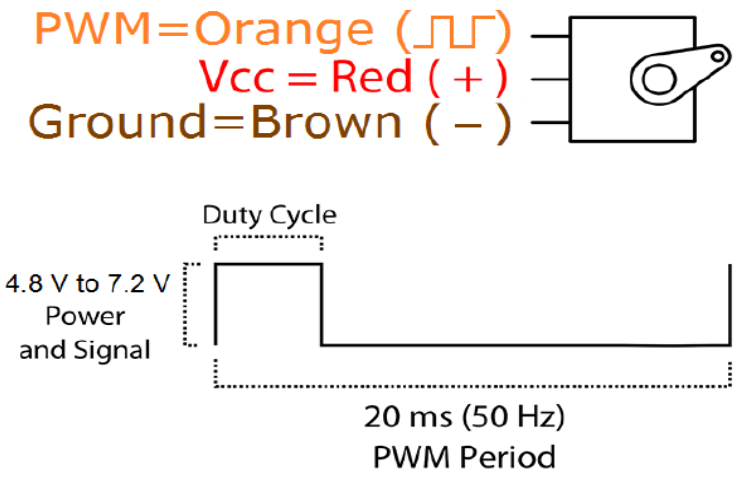
- Masa: 55 g
- Dimenzije: 40.7 x 19.7 x 42.9 mm
- Moment: 9.4 kgf·cm (4.8V), 11 kgf·cm (6V)
- Radna brzina: 0.17 s/60° (4.8V), 0.14 s/60°
- Radni napon: 4.8V do 7.2V
- Radna struja: 500 mA – 900 mA
- Struja mirovanja: 2.5A (6V)
- Stabilan i shock proof dizajn sa duplim kugličnim ležajevima
- Radna temperatura: 0 C°-55 C°

Vanjske dimenzije



Slika 4. Dimenzije elektromotora [2]

Značajke signala



Slika 5. Pinovi elektromotora i perioda PWM-a [2]

Signal slan motoru je u periodi od 20 milisekundi. Unutar tog perioda se dijeli na postotak kada je na visokom naponu i kada je na niskom. Postotak vremena unutar jedne periode u kojem je signal na naponu iznad 4.8 V se naziva „Duty cycle“. Možemo ga prikazati pomoću formule:

$$D = \frac{PW}{T}$$

Gdje D označava „Duty Cycle“, PW širinu pulsa odnosno trajanje napona iznad 4.8V unutar jedne periode i T kao oznaka za ukupno trajanje periode signala.

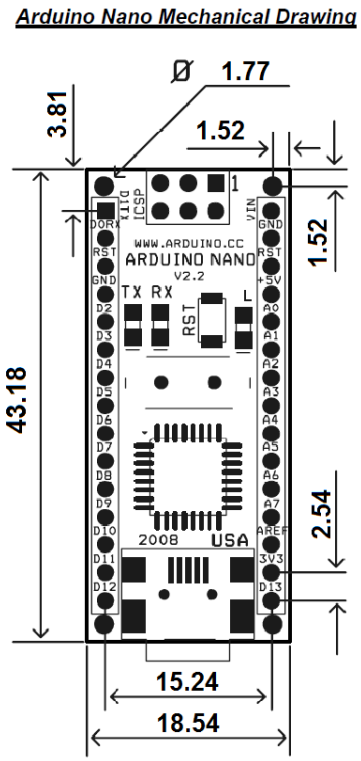
Ovisno o Duty cycle-u signala će se motor pozicionirati u pripadni kut.

2.3 Arduino Nano

Arduino Nano je mala, kompletna pločica bazirana na Atmega328P(Arduino Nano 3.x) mikroprocesoru. Napajanje dobiva preko Mini-B USB konektora. Napajenje je moguće i spajanjem na pin 30 ili pin 27. Odabran je za projekt zbog svojih malih dimenzija te zbog svojih mogućnosti.

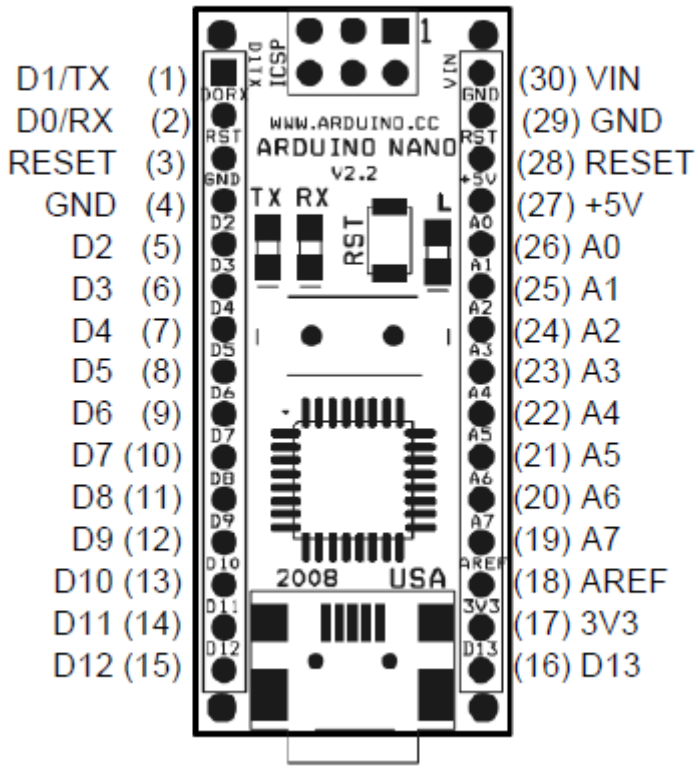
Specifikacije mikrokontrolera

- Mikrokontroler: Atmega 328P
- Radni napon: 5 V
- Ulazni napon (preporučeno): 7 V
- Ulazni napon(granice): 6-20 V
- Digitalni I/O pinovi: 14 (od kojih 6 podržava PWM izlaz)
- Analogni pinovi: 8
- DC struja po I/O pinu: 40mA
- Flash memorija: 32KB
- SRAM: 2KB
- EEPROM: 1 KB
- Brzina procesora: 16 MHz
- Dimenzije : 1.8542 x 4.318 cm
- Duljina: 43cm
- Širina 1.8cm
- Masa: 5 g



Slika 6. Dimenzije Arduino Nano-a [3]

Pinovi



Slika 7. Oznake pinova Arduino Nano-a [3]

Tablica 2 Tablica pinova za Arduino Nano

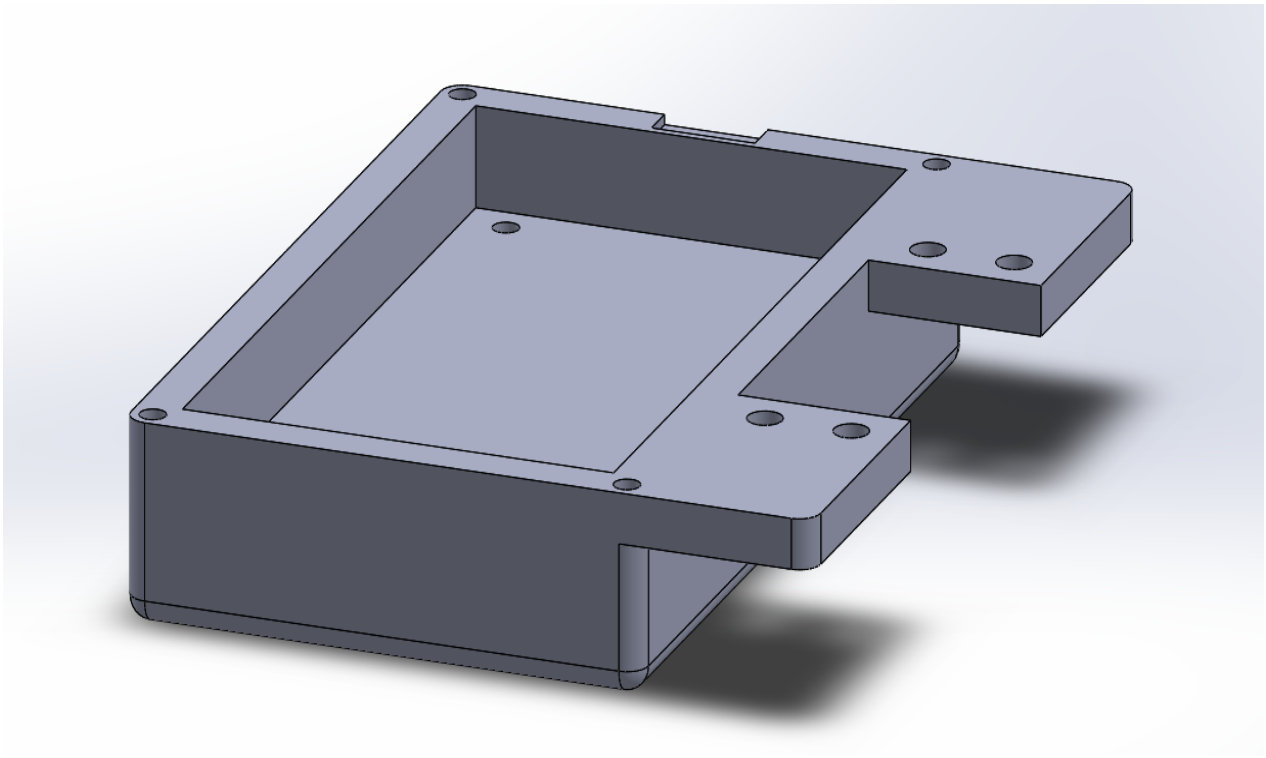
Broj pina	Ime	Tip	Opis
1-2, 5-16	D0-D13	I/O	Digitalni ulaz/izlaz port 0 do 13
3, 28	RESET	Input	Reset
4, 29	GND	PWR	Uzemljenje
17	3V3	Izlaz	+.3.V izlaz
18	AREF	Ulaz	ADC referenca
19-26	A7-A0	Ulaz	Analogni ulaz od 0 do 7
27	+5V	Izlaz ili ulaz	+5V izlaz(sa regulatora pločice) ili +5V ulaz (sa vanjskog izvora)
30	VIN	PWR	Napon

3 Konstrukcija

Konstrukcija rada se sastoji od tri dijela; postolje, poklopac i držač senzora. Te od tiskane pločice na kojoj se nalazi Arduino Nano i pinovi za povezivanje mikrokontrolera sa motorom i senzorima. Izrada konstrukcije izvedena je programskim alatom SOLIDWORKS.

3.1 Postolje

Postolje je donji dio konstrukcije unutar kojega se nalazi tiskana pločica te pričvšćen elektromotor. Dimenzije postolja su 100x80x20 mm



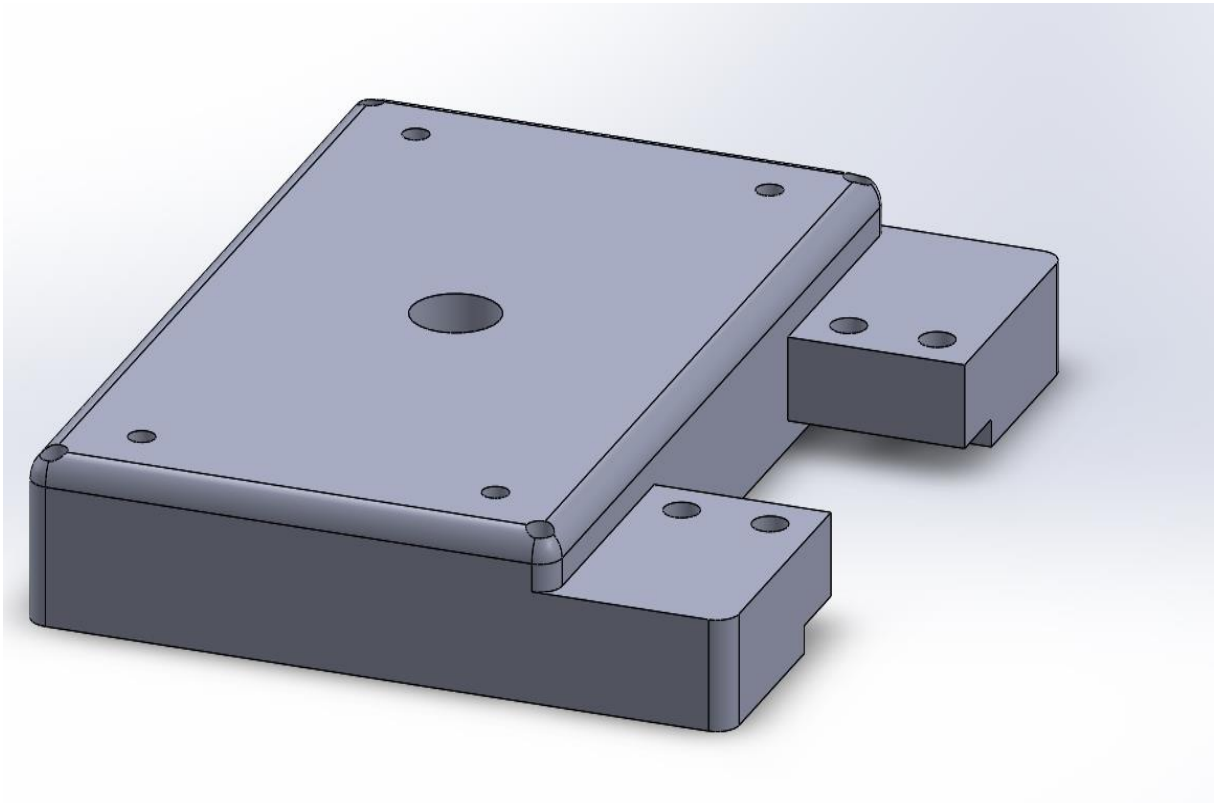
Slika 8. 3D model postolja

Samo postolje se pričvršćuje na mobilnog robota ili na nekakav proizvoljni uređaj, pomoću četiri vijka. Vijci potrebni za pričvršćivanje su M3 duljine 10 mm + duljina u koju se pričvršćuje.

Tehnički crtež se nalazi u prilogu 1.

3.2 Poklopac

Poklopac je dio konstrukcije na koji je pričvršćena tiskana pločica. Gornji dio samog uređaja je te se pričvršćuje na postolje pomoću četiri M3 vijka. Isto tako učvršćuje elektromotor.



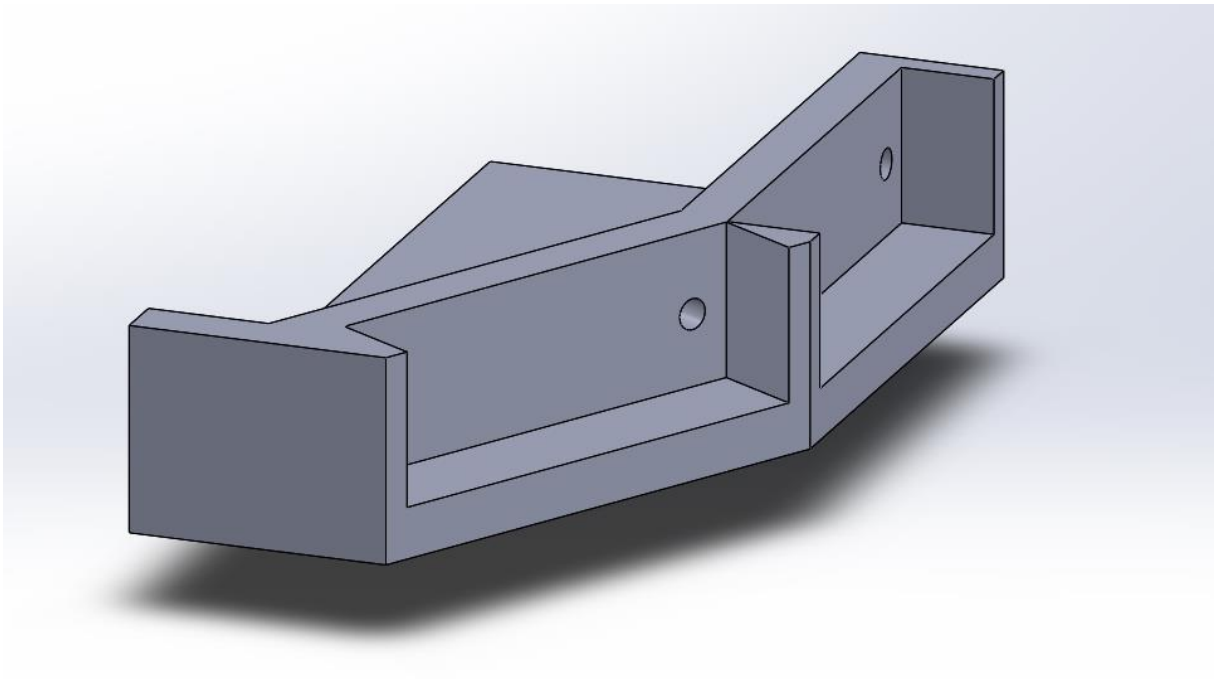
Slika 9. 3D model poklopca

Na njemu se nalazi središnji provrt za izlazne žice prema infracrvenim daljinomjerima te sa desne strane ulaz za serijsku vezu prema Arduinu.

Tehnički crtež se nalazi u prilogu 2.

3.3 Glava

Glava je dio konstrukcije koji se montira na samu osovinu elektromotora. Na njoj se nalaze dva infracrvenadaljinomjera koja su međusobno zakrenuti za 60° radi smanjenja potrebnog kuta rotacije kojim bi se zadovoljina potrebna pokrivenost skeniranog područja. Cijela glava se rotira te na taj način sam uređaj skenira područje ispred mobilnog robota.

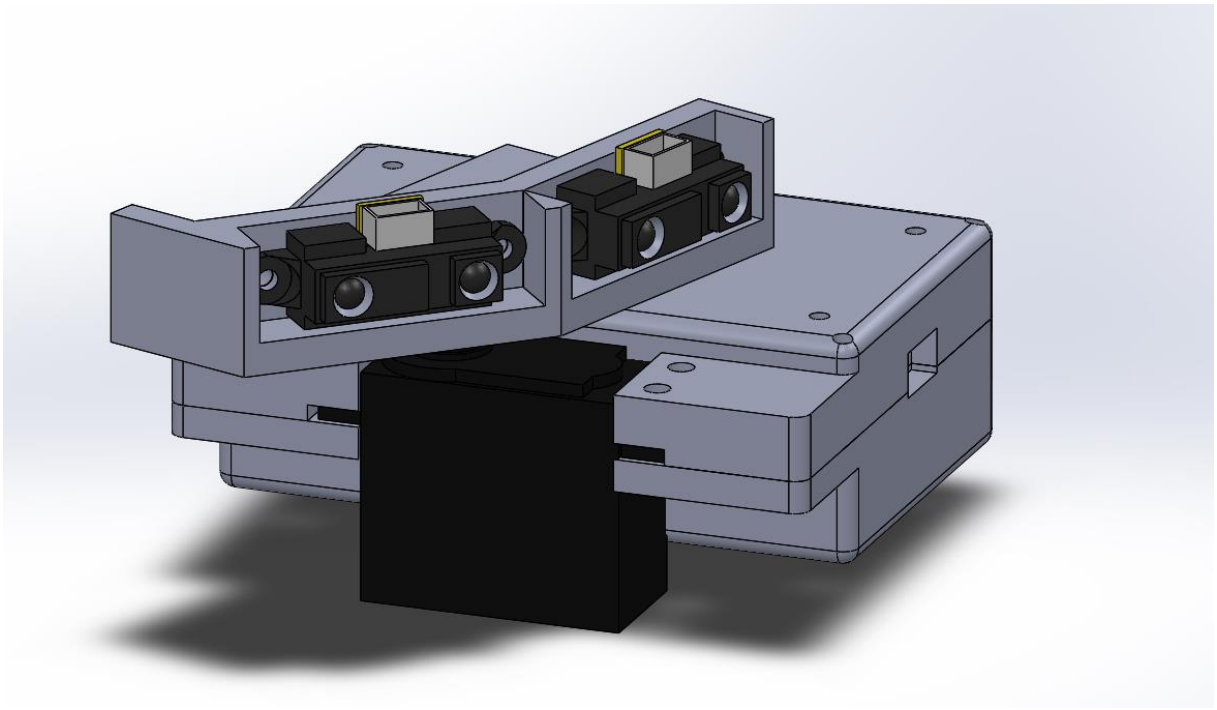


Slika 10. 3D model glave

Tehnički crtež se nalazi u prilogu 3.

3.4 Sklop

Postolje je pričvršćeno na podlogu (tijelo mobilnog robota, podloga za razna testiranja, itd.), na koje se stavlja elektromotor u za to predviđeno mjesto, prikazano crtežom. Te se u poklopac ugrađuje pločica, koji se zatim sklapa sa postoljem. Na osovinu motora se ugrađuje glava na koju su prethodno na određena mjesta postavljeni infracrveni daljinomjeri.



Slika 11. 3D model cijelog uređaja

Sklopni crtež se nalazi u prilogu 4.

4 Kod

Program se sastoji od dva programa. Prvi je pisan u Arduino programu, koji se učitava na Arduino Nano te zadaje naredbe za posao koji obavlja mikrokontroler. Drugi koji je pisan u programu Procesing te je njegov posao primljene informacije senzora prikazivati na ekranu u obliku grafa.

Prvo će biti prikazana oba koda u cijelosti, a zatim objašnjeni po djelovima.

4.1 Program za Arduino Nano modul (potpuni)

```
#include <Servo.h>
#define ir1 1
#define ir2 2
Servo myservo;
int pos = 0;
void setup()
{
  analogReference(DEFAULT);
  Serial.begin(9600);
  pinMode (ir1, INPUT);
  pinMode (ir2, INPUT);
  myservo.attach(9);
  int raw1, volt1, cm1;
  int raw2, volt2, cm2;
}
void loop()
{
   int raw1, volt1, cm1;
   int raw2, volt2, cm2;
   myservo.write(pos);
for (pos = 160; pos >=40; pos -= 1)
  {
  myservo.write(pos);
  raw1=analogRead(ir1);
  raw2=analogRead(ir2);
  volt1=map(raw1, 0, 1023, 0, 5000);
  volt2=map(raw2, 0, 1023, 0, 5000);
  cm1=(23.61/(volt1-0.1696))*1000;
  cm2=(23.61/(volt2-0.1696))*1000;
  Serial.print(pos, DEC);
    Serial.print(",");
    Serial.print(cm1, DEC);
    Serial.print(",");
    Serial.print(cm2, DEC);
    Serial.println();
    delay(15);
  }
  for (pos = 40; pos < 160; pos += 1)
  {
  myservo.write(pos);
     //Serial.println(pos,DEC);
    raw1=analogRead(ir1);
    raw2=analogRead(ir2);
    volt1=map(raw1, 0, 1023, 0, 5000);
    volt2=map(raw2, 0, 1023, 0, 5000);
    cm1=(23.61/(volt1-0.1696))*1000;
```

```
    cm2=(23.61/(volt2-0.1696))*1000;
    Serial.print(pos, DEC);
    Serial.print(",");
    Serial.print(cm1, DEC);
    Serial.print(",");
    Serial.print(cm2, DEC);
    Serial.println();
    delay(15);
  }
}
```

4.2 Program za Arduino Nano modul (objašnjenje)

U ovom djelu definiraju se biblioteke koje će se koristit (Servo.h) te portovi za infracrvene daljinomjere. Definiraju se imena senzora *Ir1* za analogni pin 1 i *ir2* za analogni pin 2. Inicijalizira se početna pozicija 0° unutar varijable *pos*.

```
#include <Servo.h>
#define ir1 1
#define ir2 2
Servo myservo;
int pos = 0;
```

Sljedeći dio koda služi za pripremu. Serial.begin() definira serijsku vezu između mikrokontrolera i samog računala. pinMode(, INPUT) definira analogni pin *ir1* i *ir2* kao ulazne pinove. Myservo.attach() postavlja servomotor na digitalni pin 9. Zatim se nalazi inicijalizacija varijabli potrebnih za čitanje mjerenih vrijednosti iz senzora, za oba senzora.

```
void setup()
{
  analogReference(DEFAULT);
  Serial.begin(9600);
  pinMode (ir1, INPUT);
  pinMode (ir2, INPUT);
  myservo.attach(9);
  int raw1, volt1, cm1;
  int raw2, volt2, cm2;
}
```

Nakon što je mikrokontroleru definirano sve o njegovoj periferiji i komunikaciji, sljedi dio koda koji se vrti u petlji sve dok ga ne zaustavimo.

```
for (pos = 160; pos >=40; pos -= 1)
  {
  myservo.write(pos);
  raw1=analogRead(ir1);
  raw2=analogRead(ir2);
  volt1=map(raw1, 0, 1023, 0, 5000);
  volt2=map(raw2, 0, 1023, 0, 5000);
  cm1=(23.61/(volt1-0.1696))*1000;
  cm2=(23.61/(volt2-0.1696))*1000;
  Serial.print(pos, DEC);
    Serial.print(",");
    Serial.print(cm1, DEC);
```

```
    Serial.print(",");
    Serial.print(cm2, DEC);
    Serial.println();
    delay(15);
  }
```

Prvo imamo for petlju koja radi prijelaz (eng. Sweep) od pozicije 160° do pozicije 40°. Razlozi zašto baš od 160 do 40 su sljedeći; motor ima tvorničko ograničenje na 120° iako programski može 180 stupnjeva. Prijelaz je odabran od 160 prema manje zbog prirodnijeg kretanja grafa na računalu s lijeva na desno pri startanju programa.

Unutar petlje servu preko naredbe myservo.write() šaljemo poziciju pos u koju se taj motor treba postaviti. Sljedeće se u varijable *raw1* i *raw2* spremaju analogna mjerenja sa pinova *ir1* i *ir2*. Zbog zapisa mjerenja intervalu od 0 do 1023 mapiramo ih u vrijednosti napona od 0 do 5 pomoću naredbe map() te te vrijednosti spremamo u varijable *volt1* i *volt2*. Nakon što smo dobili vrijednosti mjerenja u naponu, preko karakteristike senzora povezujemo udaljenost i dobiveni napon prema formuli 23.61/(volt1-0.1696))*1000 čiji rezultat u centrimetrima spremamo u varijable *cm1* i *cm2*.

U sljedećim linijama koda, preko serial.print(); podatci preko serijske veze odlaze na računalo, u formatu : *pos, cm1, cm2* novi red. Naredba novi red služi kao indikator za kraj slanja svih podataka.

Na svakom stupnju motor se zaustavlja kako bi mogao odraditi očitavanje i slanje. Vrijeme zaustavljanja definirano je naredbom delay(15), što u ovom primjeru čeka 15 milisekundi prije nego motor prebaci u sljedeći stupanj. Kasnije će biti prikazana mjerenja za 50, 25 i 10 milisekundi po stupnju zakreta motora ($9\ \frac{s}{180°}$, $4.5\frac{s}{180°}$, $1.8\ \frac{s}{180°}$).

Nakon toga ide nova for petlja koja obavlja isti posao, samo što radi sweep glave u suprotnom smjeru te tako vraća cijeli uređaj u početnu poziciju i onda sve opet iz početka.

4.3 Program za PC modul (potpuni)

```
import processing.serial.*;
Serial myPort;

int k=0, j=0;
boolean nesto=false, prvoslanje=false;
float[] brojcek=new float[13];

float slaganje(String x)
{
  int broj=0;
  int[] pomocnik=int(split(x, ','));
  broj=pomocnik[1]+255*pomocnik[2]+(255*255)*pomocnik[3];
  if (pomocnik[0]==1) {
    broj=-1*broj;
  }
  return(broj);
}

void primanje(String x)
{

  if (x != null)
  {

    x = trim(x);
    brojcek=float(split(x, ','));
  }
  nesto=true;
}

void setup() {
  String portName = Serial.list()[1];
  myPort = new Serial(this, portName, 9600);
  myPort.bufferUntil('\n');
  size(1000, 600);
  background(15, 15, 15);
  smooth();
}


void draw() {
  noStroke();
  fill(0, 0, 0, 5);
  rect(0, 0, width, height);

  crtanje_pozadine();
  stroke(20, 250, 20);
  strokeWeight(2);

  if(brojcek[1]>=60) {brojcek[1]=60;}
  if(brojcek[1]<0) {brojcek[1]=0;}
  line(500-425*cos((160-brojcek[0])*2*PI/360), 550-425*sin((160-
brojcek[0])*2*PI/360), 500-475*cos((160-brojcek[0])*2*PI/360), 550-
475*sin((160-brojcek[0])*2*PI/360));
  stroke(255, 0, 0);
  strokeWeight(5);
```

```
  line(500, 550, 500-(475*cos((160-
brojcek[0])*2*PI/360))*(brojcek[1]/80), 550-(475*sin((160-
brojcek[0])*2*PI/360))*(brojcek[1]/80) );

  if(brojcek[2]>=60) {brojcek[2]=60;}
  if(brojcek[2]<0) {brojcek[2]=0;}
  line(500-425*cos((200-brojcek[0])*2*PI/360), 550-425*sin((200-
brojcek[0])*2*PI/360), 500-475*cos((200-brojcek[0])*2*PI/360), 550-
475*sin((200-brojcek[0])*2*PI/360));
  stroke(255, 0, 0);
  strokeWeight(5);
  line(500, 550, 500-(475*cos((200-
brojcek[0])*2*PI/360))*(brojcek[2]/80), 550-(475*sin((200-
brojcek[0])*2*PI/360))*(brojcek[2]/80) );
}

void crtanje_pozadine()
{

  noFill();
  strokeWeight(2);
  stroke(0, 200, 0);
  arc(500, 550, 950, 950, PI, TWO_PI);
  line(25, 550, 975, 550);

  strokeWeight(1);
  for (int i=0; i<6; i++)
  {
    arc(500, 550, 950-i*150, 950-i*150, PI, TWO_PI);
  }
  for (int i=0; i<180; i+=15)
  {
    line(500, 550, 500-475*cos(i*2*PI/360), 550-475*sin(i*2*PI/360));

  }
}

void serialEvent(Serial myPort) {
  if (myPort.available()>0 && prvoslanje==true)
  {
    String myString = myPort.readStringUntil('\n');
    primanje(myString);
  }
  myPort.clear();
  prvoslanje=true;
}
```

4.4 Program za PC modul (objašnjenje)

U ovom djelu koda na računalu deklariramo serijsku vezu na slobodni serijski port te imenujemo varijable k i j čije vrijednosti se postavljaju na 0. Isto tako se uvode boolean varijable *nesto* i *prvoslanje* koje su označene sa false odnosno logička laž. *Brojcek* je prazni niz od 13 vrijednosti.

```
import processing.serial.*;
Serial myPort;

int k=0, j=0;
boolean nesto=false, prvoslanje=false;
float[] brojcek=new float[13];
```

Funkcija *slaganje* broj primljen serijskom vezom u 4 bytea spaja u jednu vrijednost. Uzima 3 vrijednosti do prvog zareza koji dođe serijskom vezom te ih sklapa u sljedećem formatu. Prva vrijednost koja dolazi je 0 ili 1 te ona označava predznak broja, zatim dolazi druga vrijednost koja je ostatak dijeljenja sa 255(jednim byteom) nakon njega treći broj koji se množi sa 255, i zadnji dolazi i biva množen sa 255^2. Ti brojevi se onda zbrajaju i dobiva se originalno poslani broj.

```
float slaganje(String x)
{
  int broj=0;
  int[] pomocnik=int(split(x, ','));
  broj=pomocnik[1]+255*pomocnik[2]+(255*255)*pomocnik[3];
  if (pomocnik[0]==1) {
    broj=-1*broj;
  }
  return(broj);
}
```

Funkcija *primanje* pretvara primljeni serijski podatak u zasebna polja niza *brojcek*.

```
void primanje(String x)
{

  if (x != null)
  {

    x = trim(x);
    brojcek=float(split(x, ','));
  }
  nesto=true;
}
```

Setup je funkcija koja priprema daljnji program za rad, definira port preko kojega će se obavljati serijska veza, kojom brzinom i aktivira spremnik podataka serijske komunikacije (eng. Buffer) porta dok se ne primi simbol za novi red „\n“. To je važno jer mikrokontroler šalje pakete podataka sa završnim znakom za novi red „\n“

```
void setup() {
  String portName = Serial.list()[1];
  myPort = new Serial(this, portName, 9600);
  myPort.bufferUntil('\n');
  size(1000, 600);
  background(15, 15, 15);
  smooth();
}
```

Funkcija koja obavlja crtanje samog grafa je *draw*. Unutar ove funkcije se obavlja prvo pozivanje druge funkcije pod imenom *crtanje_pozadine* koja će biti opisana kasnije, u nastavku rada. Zadim se događa provjera varijable *brojcek[1]* za prvi senzor, ukoliko je vrijednost veća od 60 program ju postavlja na vrijednost 60. Ovaj dio programa je ubačen zbog nepreciznosti senzora u vrijednostima od 60 do 80. Isti mehanizam je napravljen za vrijednosti manje od 0, zbog toga što se greške mogu pojaviti i vratiti negativan broj, što je realno nemoguće.

Nakon toga ide crtanje linije naredbom line. Ta naredba crta liniju od točke do točke, te je zato vrijednost duljine prikazana u polarnom koordinatnom sustavu. Varijabla *brojcek[0]* označava kut u kojem se trenutno nalazi servomotor te shodno tome crta ishodište linije. Iz kojeg se ona crta linija do točke udaljenje od ishodišta za vrijednost *brojcek[1]*, varijable čija vrijednost je mjerenje senzora 1.

Ista stvar se zatim radi za senzor 2 samo sa inicijalnim pomakom za 60° zbog fizičkog razmaka senzora. Na taj način se dobila zahtjevana pokrivenost od 180° ispred uređaja.

```
void draw() {
  noStroke();
  fill(0, 0, 0, 5);
  rect(0, 0, width, height);

  crtanje_pozadine();
  stroke(20, 250, 20);
  strokeWeight(2);

  if(brojcek[1]>=60) {brojcek[1]=60;}
  if(brojcek[1]<0) {brojcek[1]=0;}
  line(500-425*cos((160-brojcek[0])*2*PI/360), 550-425*sin((160-
brojcek[0])*2*PI/360), 500-475*cos((160-brojcek[0])*2*PI/360), 550-
475*sin((160-brojcek[0])*2*PI/360));
  stroke(255, 0, 0);
  strokeWeight(5);
  line(500, 550, 500-(475*cos((160-
brojcek[0])*2*PI/360))*(brojcek[1]/80), 550-(475*sin((160-
brojcek[0])*2*PI/360))*(brojcek[1]/80) );

  if(brojcek[2]>=60) {brojcek[2]=60;}
  if(brojcek[2]<0) {brojcek[2]=0;}
  line(500-425*cos((200-brojcek[0])*2*PI/360), 550-425*sin((200-
brojcek[0])*2*PI/360), 500-475*cos((200-brojcek[0])*2*PI/360), 550-
475*sin((200-brojcek[0])*2*PI/360));
  stroke(255, 0, 0);
  strokeWeight(5);
  line(500, 550, 500-(475*cos((200-
brojcek[0])*2*PI/360))*(brojcek[2]/80), 550-(475*sin((200-
brojcek[0])*2*PI/360))*(brojcek[2]/80) );
```

```
}
```

Funkcija *crtanje_pozadine* crta statičnu pozadinu samog grafa.

```
void crtanje_pozadine()
{

  noFill();
  strokeWeight(2);
  stroke(0, 200, 0);
  arc(500, 550, 950, 950, PI, TWO_PI);
  line(25, 550, 975, 550);

  strokeWeight(1);
  for (int i=0; i<6; i++)
  {
    arc(500, 550, 950-i*150, 950-i*150, PI, TWO_PI);
  }
  for (int i=0; i<180; i+=15)
  {
    line(500, 550, 500-475*cos(i*2*PI/360), 550-475*sin(i*2*PI/360));

  }
}
```

serialEvent je funkcija koja se aktivira kada se događaju promjene na serijskom portu. Kada je port primio vrijednosti aktivira se funkcija primanje.

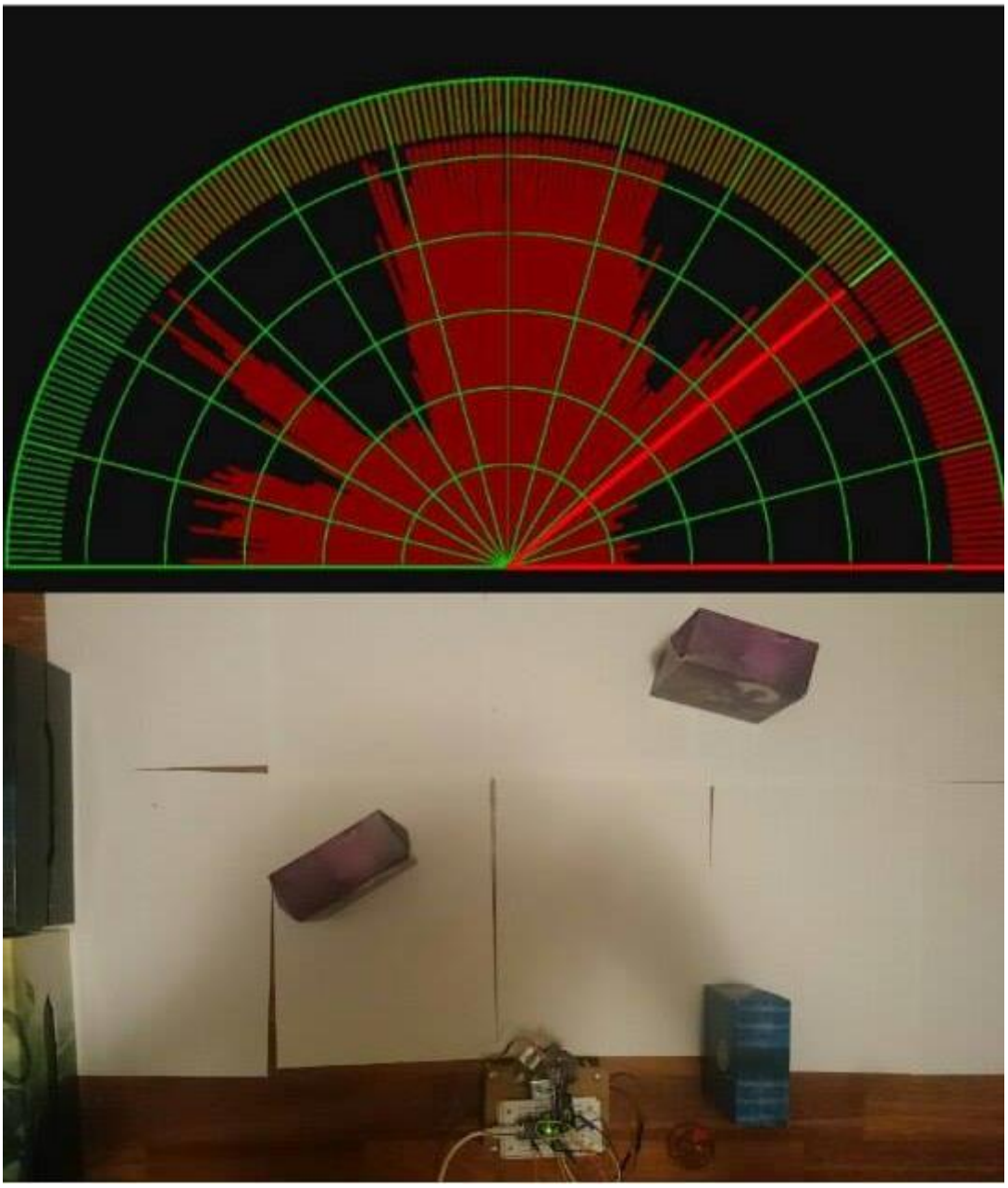
```
void serialEvent(Serial myPort) {
  if (myPort.available()>0 && prvoslanje==true)
  {
    String myString = myPort.readStringUntil('\n');
    primanje(myString);
  }
  myPort.clear();
  prvoslanje=true;
}
```

5 Usporedba mjerenja

5.1 Brzina zakreta

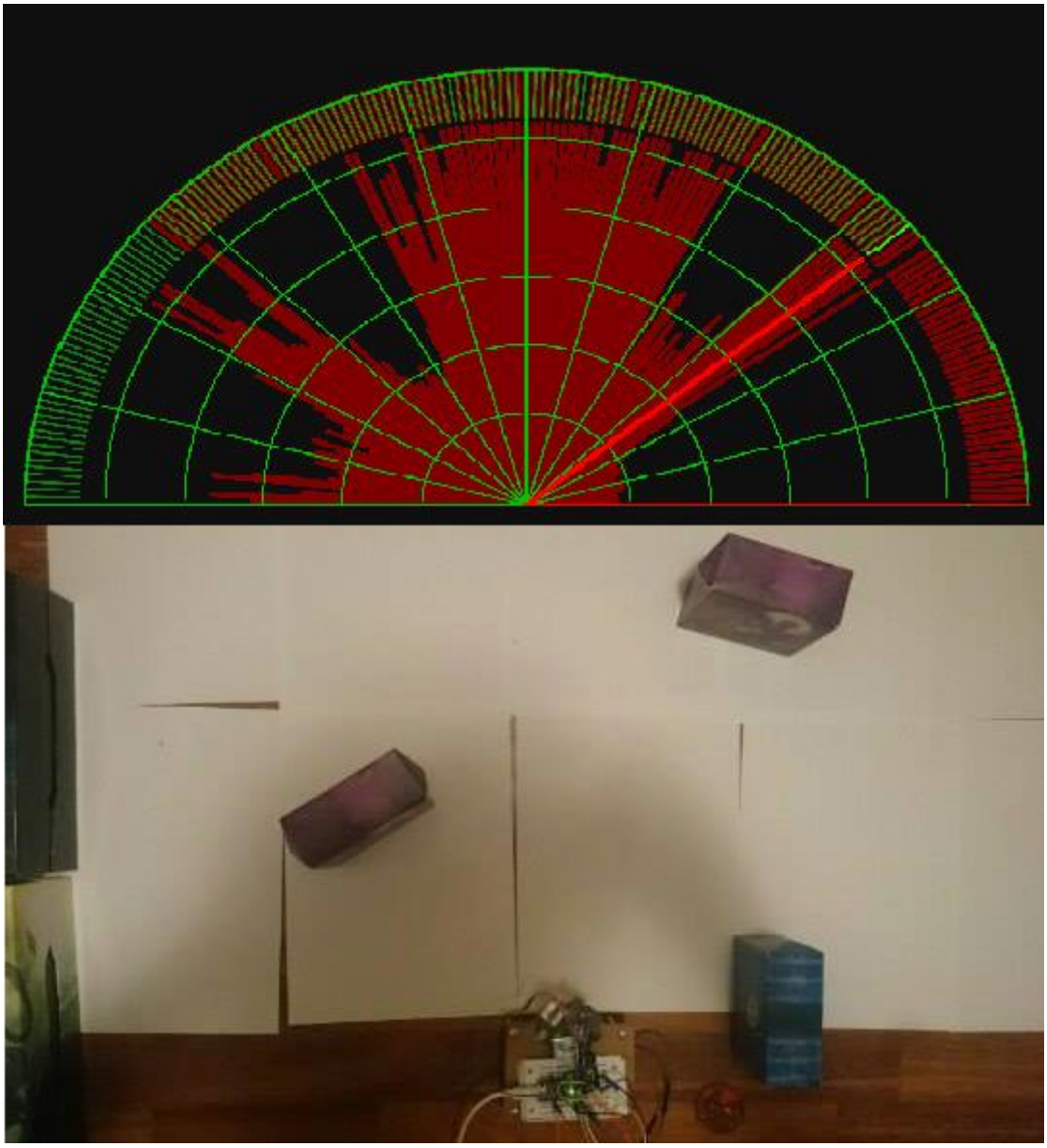
Napravljeno je mjerenje na nekoliko predmeta prethodno postavljenih na različite udaljenosti od samog uređaja, te je napravljena usporedba dobivenih podataka i postavljenih predmeta. Preciznost uređaja se gubila na predmetima udaljenijim od 60 cm, isto tako sama preciznost ovisi o brzini pokretanja servomotora, tj. o vremenu omogućenom uređaju da „gleda“ jednu udaljenost.

Prikazano mjerenje je napravljeno sa brzinom od 50 milisekundi po stupnju ($9 \frac{s}{180°}$). Iz navedenog se vidi da je mjerenje dosta precizno te da graf prikazan na računalu odgovara realnom postavu predmeta.



Slika 12. Usporedba mjerenja na 50 ms po stupnju zakreta

Prikazano mjerenje je napravljeno sa brzinom od 25 milisekundi po stupnju ($4.5 \frac{s}{180°}$). Iz navedenog se vidi da je mjerenje i dalje dosta precizno te da graf prikazan na računalu i dalje odgovara realnom postavu predmeta uz malo povećanje greške.



Slika 13. Usporedba mjerenja na 25 ms po stupnju zakreta

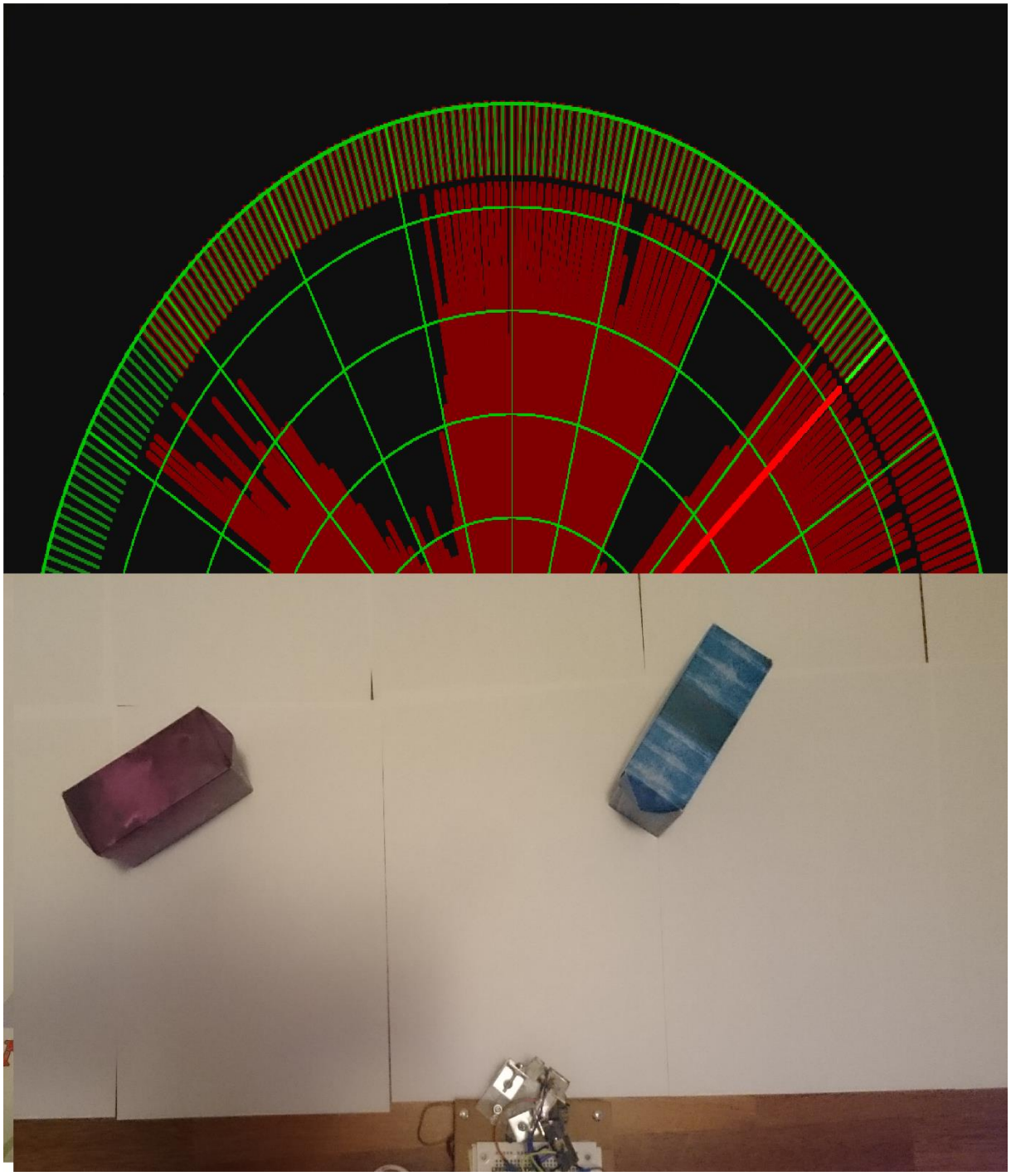
Prikazano mjerenje je napravljeno sa brzinom od 10 milisekundi po stupnju ($1.8\ \frac{s}{180°}$). Iz navedenog se vidi da je mjerenje postalo neprecizno te da se prikaz na računalu počeo razlikovati na rubovima gledanih predmeta. Te takvo mjerenje prestaje biti precizno.

Slika 14. Usporedba mjerenja na 10 ms po stupnju zakreta

5.2 Razlučivost

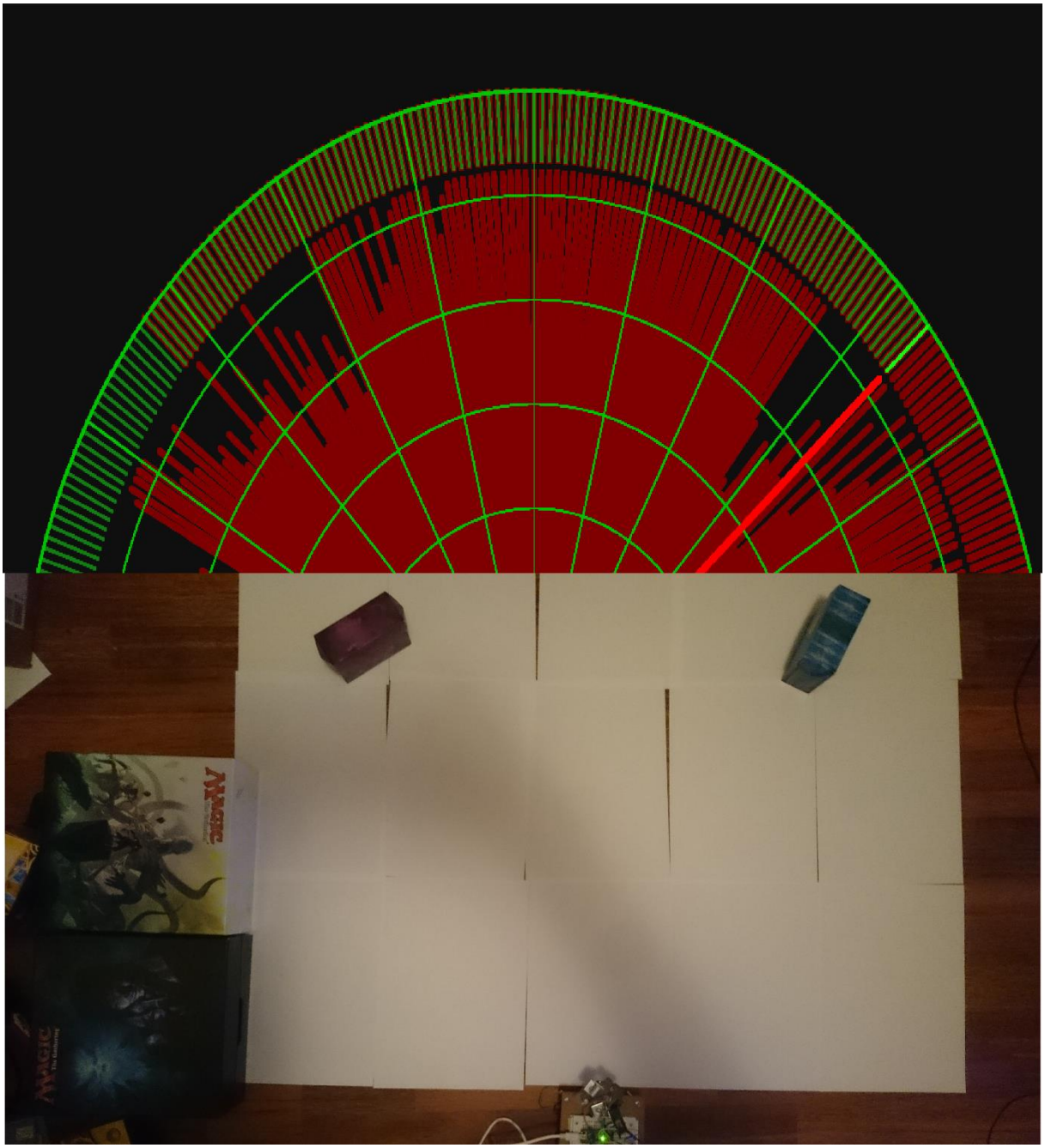
Još jedno važno svojstvo senzora je njegova razlučivost, odnostno mogućnost cijelog uređaja da na određenim udaljenostima vidi predmete raznih širina. Za mjerenje ovog svojstva su postavljena dva predmeta na jednaku udaljenost. Predmeti su različite širine. Zatim je isto mjerenje ponovljeno s istim predmetima na većoj udaljenosti.

Na prvom mjerenju oba predmeta su vidljiva na grafičkom prikazu te se može predpostaviti da je uređaj svojim „vidnim poljem“ jasno uhvatio rubove predmeta te ih i na taj način prikazao.



Slika 15. Mjerenje razlučivosti na maloj udaljenosti

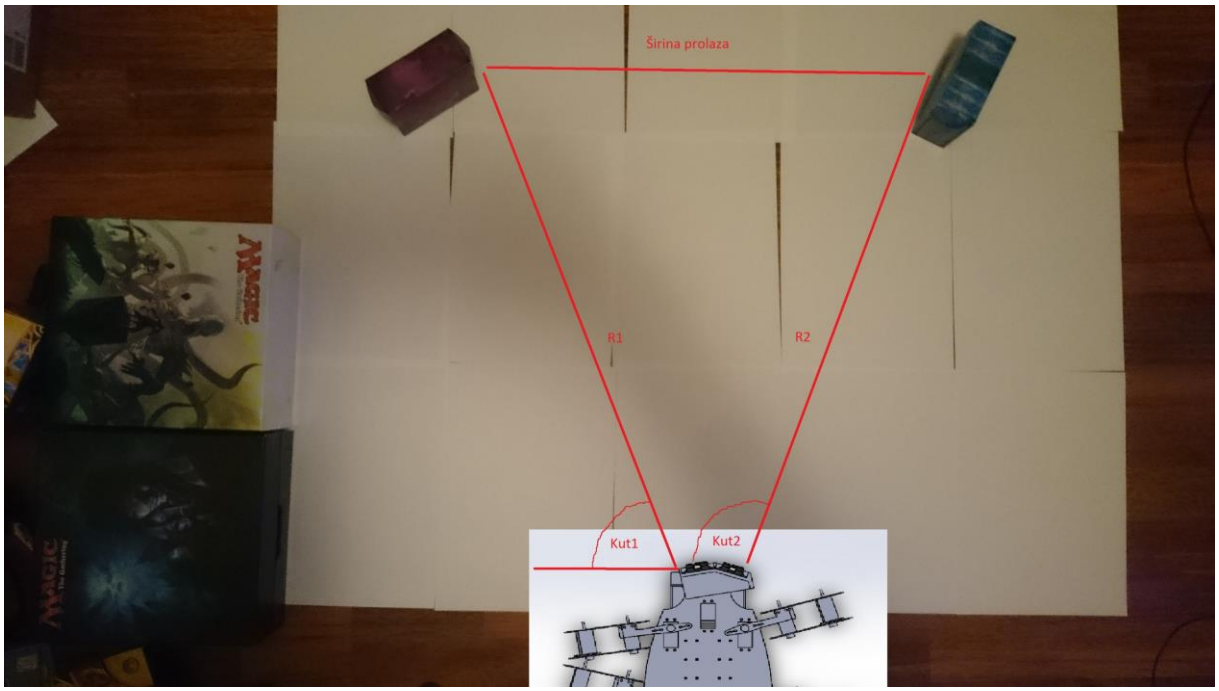
Na većoj udaljenosti predmeta, odnosno na samom maksimumu mjerne udaljenosti oba predmeta su vidljiva sa većim greškama. Rubovi mjerenih predmeta nisu više jasnih granica na grafičkom prikazu, što upućuje na smanjenje same razlučivosti uređaja. Širi predmet ima jasnije prikazan oblik, dok uži ima nejasan prikaz.



Slika 16. Mjerenje razlučivosti na velikoj udaljenosti

6 Primjer problema mobilnog robota

Nakon ugradnje uređaja na mobilnog robota potrebno je poslani signal iskoristit i prebaciti ga u informaciju korisnu za orijentaciju mobilnog robota u prostoru (izbjegavanje prepreke, prolaz između dvije prepreke). Signal koji uređaj šalje je u formatu stupanj zakreta motora, mjerenje prvog senzora i zatim mjerenje drugog senzora. Unutar tog primljenog niza možemo tražiti vrijednosti veće od maksimalne udaljenosti koju senzor može očitavati te taj niz pretvoriti u širinu preko koje možemo odrediti je li za pojedinog mobilnog robota razmak dovoljno širok za prolaz.



Slika 17. Primjer izračuna širine prolaza između dvije prepreke

Kada u prolazu uređaja od 180° dobijemo niz kutova zakreta i pripadnih mjerenih udaljenosti, možemo izdvojiti polja gdje su udaljenosti veće od maksimuma te za to polje izračunati širinu prolaza između dvije prepreke prema izrazu:

$$širina\ prolaza = 2 \cdot R1 \cdot \sin\frac{Kut2 - Kut1}{2}$$

Tada dobivenu širinu prolaza možemo usporediti sa širinom robota te na temelju toga odlučiti je li prolaz dovoljno širok za prolaz ili ne.

7 Zaključak

Ideja zadatka je za izrađeni uređaj izmjeriti ovisnost preciznosti izlaznih podataka senzora o brzini zakreta glave uređaja, udaljenosti i obliku mjerenih predmeta te prema rezultatima dobiti granice unutar kojih bi se sam uređaj mogao sa sigurnošću koristiti na mobilnom robotu.

Postepenim dizanjem brzine zakreta servomotora preciznost samog uređaja opada. Pri brzinama od 20 do 50 ms/°, preciznost je velika. Kao što se vidi iz Slike 12 mjerenje na ovakav način dobro opisuje okoliš unutar mjernog područja uređaja te bi se mogao iskoristiti na mobilnim robotima koji će imati dovoljno vremena za skeniranje prostora na ovaj način.

Naravno, ako je potrebno imati veću brzinu skeniranja, a manju preciznost samog senzora, brzina se može staviti na 15-20 ms/°, ipak za skeniranje prostora je najčešće potrebna manja preciznost (2-3 centimetra), a brže dobivanje slike o prostoru. Upravo zbog samog kretanja mobilnog robota, brzina ažuriranja podataka o prostoru je bitna.

Ukoliko se od uređaja bude tražila brzina od 10 ms/°, preciznost je jako mala. Uzroka tog opada preciznosti je tromost samog senzora, kojemu je potrebno neko vrijeme za obradu mjerenja koje je dobio. Zbog toga se može dogoditi da je servomotor već krenuo ili stigao u sljedeću poziciju, kada senzor daje podatke za prethodnu, zbog čega dobivamo kašnjenje.

Razlučivost samog uređaja je mala na udaljenostima preko 60 cm od senzora. Uz opadanje preciznosti zbog udaljenosti, opada i zbog širine gledanog predmeta. Iz primjera se vidi da predmet širine 4-5 cm ciše nije jasno vidljiv na udaljenosti od 60 cm, dok predmet širine 11 cm je, uz nejasne granice rubova predmeta.

Na primjeru gdje su oba predmeta mjerena na maloj udaljenosti od svega ~10 cm granice oba predmeta su jasno vidljiva neovisno o njihovoj širini.

8 Literatura

[1] SHARP GP2Y0A21YK0F data sheet
https://www.pololu.com/file/download/gp2y0a21yk0f.pdf?file_id=0J85

[2] MG996R_Tower-Pro data sheet
http://www.electronicoscaldas.com/datasheet/MG996R_Tower-Pro.pdf

[3] ArduinoNanoManual23
https://www.arduino.cc/en/uploads/Main/ArduinoNanoManual23.pdf

[4] PEPPERL+FUCHS OMT100-R100-EP
https://files.pepperl-fuchs.com/webcat/navi/productInfo/doct/tdoct4997c_eng.pdf?v=20160623134154

[5] PEPPERL+FUCHS VDM18-300/21/122/151
https://files.pepperl-fuchs.com/webcat/navi/productInfo/doct/tdoct1314a_eng.pdf?v=20110817132029

[6] PARALLAX Ultrasonic Distance Sensor #28015
https://www.parallax.com/sites/default/files/downloads/28015-PING-Sensor-Product-Guide-v2.0.pdf

9 Prilog

Prilog 1 (Postolje)
Prilog 2 (Poklopac)
Prilog 3 (Glava)
Prilog 4 (Skener)